Short: Device-to-Identity Linking Attack Using Targeted Wi-Fi Geolocation Spoofing

Célestin Matte, Jagdish Prasad Achara, Mathieu Cunche

Short: Device-to-Identity Linking Attack Using Targeted Wi-Fi Geolocation Spoofing*

Célestin Matte
University of Lyon, Inria
celestin.matte@insa-lyon.fr

Jagdish Prasad Achara
Inria
jagdish.achara@inria.fr

Mathieu Cunche
University of Lyon, Inria
mathieu.cunche@inria.fr

ABSTRACT

Today, almost all mobile devices come equipped with Wi-Fi technology. Therefore, it is essential to thoroughly study the privacy risks associated with this technology. Recent works have shown that some Personally Identifiable Information (PII) can be obtained from the radio signals emitted by Wi-Fi equipped devices. However, most of the times, the identity of the subject of those pieces of information remains unknown and the Wi-Fi MAC address of the device is the only available identifier. In this paper, we show that it is possible for an attacker to get the identity of the subject.

The attack presented in this paper leverages the geolocation information published on some geotagged services, such as Twitter, and exploits the fact that geolocation information obtained through Wi-Fi-based Positioning System (WPS) can be easily manipulated. We show that geolocation manipulation can be targeted to a single device, and in most cases, it is not necessary to jam real Wi-Fi access points (APs) to mount a successful attack on WPS.

Categories and Subject Descriptors

K.4 [**Public Policy Issues**]: Privacy

General Terms

Security; Privacy; 802.11; Geolocation

1. INTRODUCTION

Wi-Fi technology has become very commonplace today and is being increasingly deployed in almost all kinds of mobile devices. As these mobile devices need to connect to a Wi-Fi AP to access the Internet, they often have their Wi-Fi interface enabled. For active service discovery, these Wi-Fi-enabled devices periodically broadcast probe requests containing the MAC address of their Wi-Fi interface. This raises serious privacy risks to the user as MAC address is a unique identifier and can be used to track the device [8].

However, the privacy threats are limited if only device-unique identifier, i.e., the Wi-Fi MAC address, is known to the attacker *and not the identity of the user*. By "identity", we mean any kind of information that can help an attacker to identify the target, including their real name or any of their online profiles. Mobile applications might have access to both the user identity and the Wi-Fi MAC address of the device [2]. However, such information is generally not available to a physical proximity-based attacker.

Contributions. In this paper, we present an attack that leads an attacker, in target's physical proximity, to find identity information about the target after having obtained the target's MAC address as described in [3]. The attack exploits a limitation of Wi-Fi-based Positioning Systems (WPSs) in order to spoof the geolocation of the user. This spoofed geolocation is then used as a side-channel information source to establish the link between a Wi-Fi MAC address and the user's profile on a geotagged service. As the attacker could be interested in getting identity information about a group of people or a single acquintance, the geolocation spoofing must differ accordingly. In case of a single acquaintance, the attacker must spoof the geolocation of this acquintance's device only. We show how an attacker can spoof the location of a single device without affecting other devices in range. Finally, unlike previous work [13], our experiments reveal that location spoofing in WPSs can be done *only* by creating fake Wi-Fi APs. In most cases, jamming real Wi-Fi APs is not necessary and thus, the attack would go unnoticed by other Wi-Fi users.

Outline. In section 2, we discuss privacy issues associated to Wi-Fi and present details about Wi-Fi-based Positioning Systems (WPSs) as well as location services in mobile OSs. Section 3 covers the related work. Section 4 presents the targeted geolocation spoofing attack and section 5 describes its application to the *device-to-identity linking* attack. Experimental results are presented in section 6, and section 7 concludes the paper.

2. BACKGROUND

Wi-Fi & Privacy.

In order to discover surrounding APs, Wi-Fi-enabled devices employ either *active* or *passive* service discovery mode. In the active mode case, Wi-Fi-enabled devices broadcast frames known as *probe requests*, to which surrounding APs reply with a *probe response*. Those probe requests might

*This work is partially funded by Région Rhône-Alpes's ARC7 and Inria project lab CAPPRIS.

contain the names (Service Set Identifiers or SSIDs) of previously connected networks. In the passive mode case, devices passively listen to *beacons*, broadcast by APs to announce the characteristics of the corresponding Wi-Fi network.

Active service discovery is generally employed by mobile devices, because it is less energy-consuming and faster than the passive one. Previous works have shown that it is possible to infer a lot of personal information from SSIDs contained in the probe requests emitted by Wi-Fi enabled devices [6, 4]. Keeping in mind the privacy threats caused by it, major vendors started to broadcast probe requests containing no SSID and Wi-Fi APs are obliged to respond to all broadcast probe requests. This initiative reduced the privacy risks, but these periodically broadcast (every 20-30 seconds in the case of mobile devices) probe requests still contain the MAC address of the Wi-Fi interface. Thanks to this unique identifier, tracking the device, and thus its owner, is trivial [8]. This technique is used by retail stores [1] to implement physical analytics and advertisement campaigns, but can also be used by any curious eavesdropper.

Wi-Fi-based Positioning System (WPS).

Like any other Wi-Fi enabled device, Wi-Fi APs are also uniquely identified by their MAC address, called Basic Service Set Identifier (BSSID). As Wi-Fi APs are usually static in nature, they can be used as landmarks to create a geolocation system. Thus, given a set of visible APs along with their received signal strength (RSSI), it is possible to compute the geolocation of the device either using a trilateration [10] or a fingerprinting [15] technique. Today, several actors, such as Google, Apple, Skyhook, Navizon or Mozilla provide WPS services.

Location Services in Mobile OSs.

As Android and iOS are the two most widely used mobile OSs today, we look further in details how geolocation is derived in these two OSs. In fact, these OSs provide a system service to which applications can ask for geolocation of the device. The dedicated service in both OSs normally geolocates the user based on all available input sensors, i.e., GPS and information related to Wi-Fi, cellular and (sometimes) Bluetooth networks. However, in places where GPS sensor is not available or if the device is not equipped with such a sensor, this service will geolocate the user solely based on other sources, e.g., Wi-Fi and cellular. Furthermore, Wi-Fi is the only source for this service if the device does not have a cellular connection. We should also note that Wi-Fi information is often preferred to cellular information as it provides a more precise geolocation. These system location services are generally pre-configured to query their respective APIs (Apple for iOS and Google for Android).

3. RELATED WORK

Manipulation of a Wi-Fi positioning system has been studied in [13]. The authors showed that it is possible to arbitrarily change the geolocation information of all devices in range by jamming local APs (injection of white noise on some channels) and creating fake Wi-Fi APs. However, our experiments in section 6 reveal that jamming is not always necessary and geolocation spoofing can be achieved only by creating fake Wi-Fi APs. Also, the attack presented in [13] will affect all devices in range, whereas the attack presented in this paper allows the attacker to target a specific device based on its MAC address. It is probably worth mentioning that targeting a specific device is needed if the attacker wants to get the identity of a single person.

The problem of linking a Radio-Frequency (RF) device to a person's identity has been studied in [9]. This work demonstrates how RF and visual information can be combined to infer the link between the MAC address of a Wi-Fi device and the person's visual identity (visual characteristics such as clothes). Correlation between received signal strength (RSSI) variations and movements are exploited to accurately link a physical moving object to the source of a radio signal. In this work, devices are also identified by their MAC address, but instead of the visual identity, we focus in this paper on the digital identity of the device owner.

The idea of using a side channel to identify individuals associated to some raw data such as mobility traces has been discussed in [12]. The authors use a social network to de-anonymize mobility traces. They exploit the correlation between the contact graph obtained from the mobility traces with the social graph from the social network. In our work, the geolocation information is used as the side channel conveying information for the identification of the individual associated to a wireless device.

Privacy issues associated to geotagged information posted on online platforms such as Facebook or Google+ have been highlighted in [7]. This work focuses on privacy issues caused by photos uploaded by other users and studies how location information embedded within them can be harmful. In fact, these services revealing publicly geolocation information could be exploited in the device-to-identity-linking attack presented in section 5.

4. TARGETED WI-FI GEOLOCATION SPOOFING

In this section, we introduce a variation of the WPS spoofing attack presented by Tippenhauer et al. in [13]. Rather than affecting all devices as in [13], we show how one can spoof geolocation of only one device in range. Below, we first describe Wi-Fi geolocation spoofing, then we present its targeted version.

4.1 Wi-Fi Geolocation Spoofing

WPSs rely on BSSIDs of surrounding Wi-Fi APs for geolocation. As a result, to spoof the location, it is enough to create fake APs that are known to exist in another location. Indeed, it is not necessary to implement fully functional APs as the WPS only uses the result of Wi-Fi scans. Therefore, it is sufficient to only implement the advertising functionality of the AP, i.e., the two service discovery mechanisms:

- Emitting beacons with the MAC address (BSSID) and name (SSID) of Wi-Fi APs,
- Replying to probe requests with probe responses containing (B)SSIDs of desired Wi-Fi AP.

It is possible to create a set of fake Wi-Fi APs with a single computer. In fact, all we need is a Wi-Fi interface supporting monitor mode and packet injection. Nowadays, most commercial Wi-Fi cards are capable of doing this with readily available software like aircrack-ng[1], provided compatible drivers exist.

[1] `http://www.aircrack-ng.org/`

4.2 Targeted Spoofing

As already mentioned, in passive service discovery mode, Wi-Fi APs send beacon frames. These frames are by convention sent with a broadcast destination MAC address, and thus, should be processed by all devices in range. However, destination address of beacons can be set to the MAC address of a given device. By doing so, those frames would only be processed by the targeted device, and thus, would be ignored by others. This is possible because filtering packets depending on the destination MAC address is done by the Wi-Fi driver on the 802.11 layer (as well as by the kernel, as detailed in [11]). Upper-layer OS services or applications will not check if that field is coherent for the received packet, i.e., check if beacons have a broadcast destination address.

In active service discovery mode, Wi-Fi APs respond to all probe requests independently from the address of the requesting station. However, it is possible to modify an AP such that it only responds to probe requests coming from a given MAC address.

Therefore, to be able to create a fake AP only visible by a device of Wi-Fi MAC address `A`, it is possible to target both service discovery modes (active and passive):

- Sending beacon frames with destination MAC address set to `A` (Passive service discovery)
- Only responding to probe requests coming from MAC address `A` (Active service discovery)

5. DEVICE-TO-IDENTITY LINKING ATTACK

This section describes the Device-to-Identity Linking attack in detail. The objective of this attack is to link the Wi-Fi MAC address of a device to its owner's digital identity. Collecting the list of online accounts related to a person without his consent may be interesting for many reasons. For instance, a malicious person willing to know more about an acquaintance may launch the attack after having obtained the MAC address [3]. In the context of commercial Wi-Fi tracking systems, identifying this link could be used to increase the amount of information collected for profiling or targeted ad delivery. Similarly, in the case of a surveillance system using Wi-Fi tracking, this link could be used to identify an individual or a group of persons in a demonstration or any crowded event.

5.1 Attacker Model

First, the attack requires that the attacker is physically close to the target to be able to inject arbitrary Wi-Fi traffic. Second, we assume that the attacker has access to online profiles where geolocation information is available. This may be due to the fact that the geolocation information is publicly available, e.g., geolocation-enabled tweets on Twitter or because the attacker has privileged access to the data of a geotagged platform, e.g., Google+, Tinder or Waze. In the first case, as the information is publicly available, any attacker can simply query or crawl the geotagged platform whereas the latter case corresponds to a more powerful attacker such as an employee of a company producing such apps or a law enforcement or surveillance agency.

5.2 Details of the attack

Below are the steps an attacker needs to follow to successfully perform the attack.

1. Selecting the destination location for spoofing.

The first step of the attack is the selection of the location to where the location will be changed, i.e. the destination location. To improve the chances of success of the attack, the attacker must choose a destination location with a comparatively high number of Wi-Fi APs and a low number of people. The best way to do this is to build a dedicated Wi-Fi environment in a remote uninhabited area. However, this requires the attacker to be highly motivated, resourceful and prepared well in advance. For a more modest attacker, such areas could be residential areas during the day and business or industrial areas at night. When selecting the destination, the attacker also has to make sure that he has enough Wi-Fi APs because the number of fake APs should be higher than the number of actual ones for geolocation spoofing to work (See section 6.1). This verification can be done with the aid of online Wi-Fi registries, such as WiGLE[2], that provide a map displaying available APs in a given location.

2. Getting list of Wi-Fi APs in destination location.

Once the destination location is selected, the attacker needs to retrieve the BSSIDs of the APs found at this location. This is already known to the attacker if he has built his own dedicated Wi-Fi environment in a remote uninhabited area. For a normal attacker, this information can once again be obtained from an online Wi-Fi registry such as WiGLE. WiGLE allows precise queries and provides exhaustive information about Wi-Fi APs such as their precise location, BSSID, operating channel, etc.

3. Estimating the spoofed location.

Knowing the exact position where the target will be "*teleported*" is necessary to distinguish them from other users that may be found around the destination location. An estimation of this position can be obtained by querying the appropriate WPS through its API. This estimation increases the chance of success of the attack, because with the same input list of Wi-Fi APs, geolocation returned by WiGLE and the target WPS might differ, due to differences in their position estimation algorithms and database of APs.

4. Monitoring geotagged updates.

After obtaining the exact spoofed location from the previous step, the attacker can start monitoring the geotagged platform for updates coming from that specific location. The feasibility to monitor geotagged platforms depends on the attacker's capabilities as well as the platform. For instance, in the case of Twitter, it is easy to monitor new tweets from a specific location because Twitter provides a "streaming API"[3] that contains a method to monitor new tweets based on a geographic criteria. Therefore, this service offered by Twitter could be exploited by any random attacker, irrespective of their capabilities. Similarly, Facebook also offers a graph search API[4] where users can search for public ob-

[2] `https://wigle.net/`

[3] `https://dev.twitter.com/streaming/overview/request-parameters#locations`

[4] `https://developers.facebook.com/docs/graph-`

jects of information by location. Instagram, another popular photo/video sharing social network, offers an API[5] to get a list of recent media objects from a given location. More generally, [5] proposes a method to gather data based on location from social media websites.

5. Spoofing the location.

Once the monitoring of the geotagged platform is set up, the attacker can launch the core of the attack by impersonating the Wi-Fi APs obtained for the destination location using techniques described in section 4. This geolocation spoofing can be either targeted or not, depending on the attacker's needs. For example, if the attacker wants to get the identities of all the people in a gathering, he would not need to perform targeted geolocation spoofing, whereas he would have to if he is interested in the owner of a specific device.

5.3 Limitations

For this attack to work, the target must use a geotagged services during the attack. If the target of the attacker is a group of people, i.e., the attacker wants to know identities of people present in a gathering, it is probable that some of them will be using one of the geotagged services the attacker has access to. However, if the attacker is interested in a single acquaintance, the attacker might need to stay longer in contact with the target for the attack to be successful. Monitoring multiple geotagged platforms could again increase the chances of a successful attack.

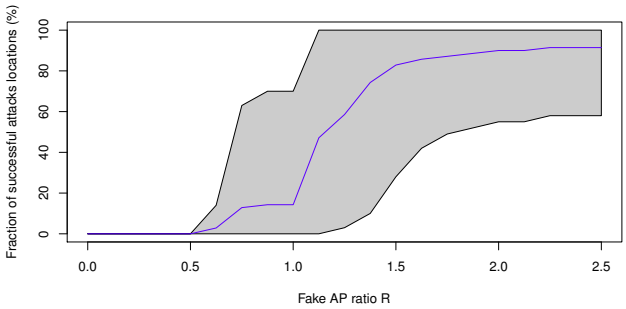
6. EXPERIMENTS AND RESULTS

To validate and test the feasibility of the attack described in the previous section, we conducted various experiments. In this section, we describe these experiments and present our findings.

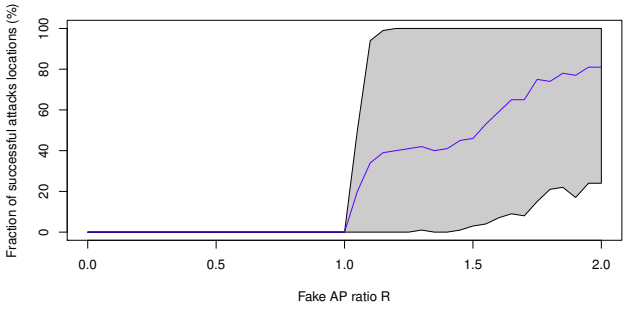
6.1 Geolocation spoofing and WPSs

We evaluated the effect of geolocation spoofing on multiple WPSs. Jamming disrupts Wi-Fi networks for all devices in range and can thus be quickly detected. Reactive jamming is not an option, as beacons and probe responses are too short to be selectively jammed with 100% accuracy [14]. Therefore, we considered the possibility of performing the spoofing attack without *erasing* the local APs (as opposed to [13]).

The following WPSs have been tested in our experiments: GoogleGeoloc[6], Navizon[7] and Skyhook[8]. The protocol used to evaluate the effect of geolocation spoofing on different WPSs is: for each pair of locations, the *origin* and the *destination*, we consider a spoofing attack from the origin to the destination. The outcome of the spoofing attack is simulated by submitting a list of Wi-Fi APs to the WPS's API. This list of Wi-Fi APs contains only APs from the origin location and the testing protocol proceeds by iteratively adding APs from the destination locations. In the first iteration, a request is sent with only APs from the origin. In the next iteration, one AP from the destination is randomly selected and added to the request. This process is repeated until a predefined ratio of fake APs over real APs is reached.



(a) Google Geolocation API



(b) SkyHook Geolocation API

Figure 1: Fraction of successful attacks as a function of the fake AP ratio R (average, 5th and 95th percentiles)

Figure 1 presents the evolution of the geolocation returned by the WPS as a function of the ratio of APs from destination to original location, i.e., $R = N_{\text{dest}}/N_{\text{orig}}$, for Google and Skyhook APIs. We tested Navizon as well, but it takes history of previous locations into account, making it more resistant against our attack. For Google and SkyHook APIs, a clear change in position can be observed around the value of 1 for the APs ratio. It appears that WPS relies on a majority vote decision to resolve conflicts when APs from two clearly distinct locations are submitted in the same request. The returned location is the one that corresponds to the location where a larger number of APs are present in the request. When the numbers of APs from the distinct location are close, the outcome is less predictable.

The result of our experiments shows that WPS spoofing can be performed on these WPSs without jamming as long as the fake APs are in larger number than the local APs. For this reason, WPS spoofing attack must use a destination location where the number of APs is larger than at the original location.

6.2 Firefox location API: a case study

HTML5 browsers provide an API for websites to geolocate the device[9]. We looked at the implementation of this API in an open source browser (Firefox). Below are our findings.

When a geolocation request is made by a web application, Firefox prompts the user for the permission to share the ge-

api/using-graph-api/v2.1#search

[5] https://instagram.com/developer/endpoints/locations/#get_locations_media_recent

[6] https://developers.google.com/maps/documentation/business/geolocation/

[7] https://www.navizon.com/wifi-cell-tower-location-database

[8] https://my.skyhookwireless.com/resources/view/precisionlocation/linux/quickstart

[9] getCurrentPosition()

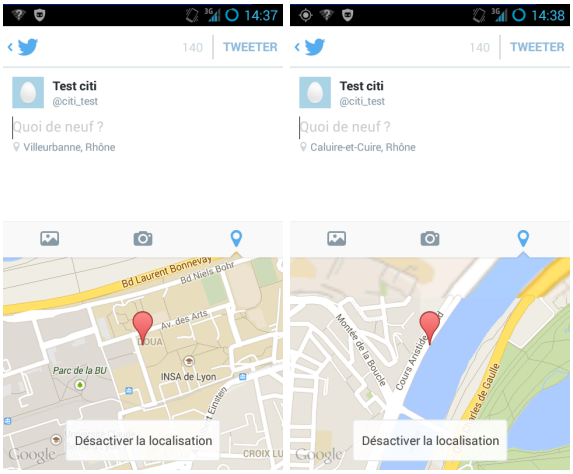


Figure 2: The Twitter application, before and during the attack.

olocation. If allowed by the user, Firefox decides whether it can geolocate the device using the available cache. If not, it asks one of the available geolocation providers. The chosen provider depends on the system: for example, Android uses Android location provider. On some systems like OSs running on laptops or desktop computers (Linux, Windows...), no specific system location provider is available. In this case, Firefox will use its default network-based location provider, which currently performs the queries to Google by default. This default network-based location provider uses information from Wi-Fi and cellular sources to derive the approximate user location. In the case of laptops and desktop computers, as there is no SIM card, the only available source to geolocate is Wi-Fi (assuming the device has one Wi-Fi interface). We discuss below how this Wi-Fi-based geolocation is derived in Firefox.

When a geolocation request is made by a website and if the user agrees to share the geolocation information, it asks the Wi-Fi subsystem to scan the Wi-Fi APs and notify the changes in the list of surrounding APs. From this returned list of surrounding Wi-Fis, it first filters out the Wi-Fi APs whose SSID field contains "_nomap". Remaining surrounding APs are sorted based on their signal strength in decreasing order. This new list of APs is then cross-checked with the cached location requests. A cached location request contains the list of Wi-Fi APs seen when that request was done and the geolocation coordinates derived from that list. If more than 50% of APs match between the list of APs stored in last cached location request and the current list of surrounding APs, the stored location coordinates in the last request are returned. If the matched APs are less than 50%, a new geolocation request is sent to Google location API.

To confirm if device-to-identity linking attack is feasible on a target using Firefox browser on a laptop, we performed our attack on targets running Mac OS X and Linux at location X. By creating equal number of fake APs from another location Y, we were able to spoof the geolocation of the user to the location Y.

6.3 A Proof-of-concept of the attack

Our proof of concept of the device-to-identity linking attack is demonstrated on Android and iOS. We performed our tests on two Android devices: a Wiko Rainbow running Android 4.2.2 and a Nexus S running CyanogenMod 10.2.1. Both produced similar results. The iPhone device was a iPhone 5S running iOS 8.1.3. As a source of side-channel information, we used the Twitter application (version 5.2.4) on Android and iOS. The location services used in the experiments are from Google and Apple on Android and iOS respectively. The environment used for the test contained 30 legitimate APs in the 2.4 Ghz band, and we generated two times that number of fake APs. The phone was two meters away from the computer running the attack.

In addition to a Wi-Fi interface supporting monitor mode and packet injection, our attack requires a WiGLE account, a Google API key to use their respective APIs, and a patched version of aircrack-ng or mdk3[10] to create fake Wi-Fi APs. We developed a tool that automatically performs all the steps necessary for the attack as described in section 5.2. The program is written in Bash, Perl and PHP, and is available on GitHub[11]. This tool takes as input the Wi-Fi MAC address of the targeted device and the desired destination location. As described in section 4, this tool performs Wi-Fi impersonation in both active and passive mode. Replying with targeted probe responses to broadcast probe requests is done with a modified version of aircrack-ng while the generation of targeted beacons is done with mdk3.

Figure 2 shows the Twitter Android application before and during an attack. The target was moved by more than 1km with an accuracy at the destination of roughly 100m. This would have led an attacker to directly identify the Twitter profile of the targeted device's owner. On iOS, we could not successfully perform the attack. Since the internals of iOS are not public, we can only speculate on the possible reasons for this: location cache containing more precise location information or spoofing countermeasures.

Table 1: Results of the Wi-Fi geolocation spoofing on selected Android applications. Second column indicates whether the geolocation is public and therefore does not require a privileged access to the data.

Application name	public geolocation	Result of geolocation spoofing	
		GPS off	GPS turned on, then off
Messenger (Facebook)	✗	✗	✗
Facebook	✗	✓	✗
Twitter	✓	✓	✓
Google+	✓	✓	✓
Foursquare	✓	✓	✗
Swarm	✗	✓	✗
Instagram	✓	✗	✗
Tinder	✓	✓	✓
Badoo	✓	✓	✓
LOVOO	✓	✓	✓
RunKeeper	✗	✗	✗
Nike+ Running	✗	✗	✗
Waze	✓	✗	✗
Glympse	✓	✓	✓
Glympse Express	✓	✓**	✓**
Runtastic	✗	✓	✓

**: only if the attack is launched beforehand

6.4 Impact of geolocation spoofing on mobile applications

As applications on Android might use alternate source of geolocation information [2] or use a dedicated cache, they

[10] http://aspj.aircrack-ng.org/#mdk3
[11] https://github.com/Perdu/geoloc_attack

might react differently to our location spoofing attack. In order to evaluate the applicability of this attack on other geotagged platform, we studied the impact of geolocation spoofing on the corresponding Android applications.

We selected 16 Android applications that extensively geolocate the user and allow this information to be published by the platform. These mobile applications may or may not make the geolocation information publicly available but all of them send the geolocation information to their servers. Thus, in cases where an attacker has a privileged access to this information, he will be able to launch a device-to-identity linking attack if the geolocation used by these applications can be manipulated by Wi-Fi spoofing. We performed these tests in an environment containing 20 legitimate APs, and we generated 50 APs for each test. We marked the attack as successful when the target was effectively moved near the target location, after any period of time. We sometimes had to close and re-open the application, as the attack did not work at the first attempt.

The results of the experiments are presented in Table 1 and demonstrate that a good fraction of applications are vulnerable to geolocation spoofing. Different test configurations have been evaluated, which yielded different results. If the GPS was activated and had acquired a location, the attack then failed for all apps. However, if it was activated but did not receive a signal, the results were the same as when the GPS was disabled, except for the fact that some apps would not complain that the GPS was not activated. Running the tests after the GPS was disabled yielded different results from running the tests with GPS disabled at all times: some apps kept the (actual) location provided by the previously activated GPS, while others trusted the new fake geolocation immediately (see the third column of Table 1).

We also tested some other potential factors, which did not happen to have any influence on the results of geolocation on Android: for example, being associated or not to an access point turned out to have no impact, and providing fake SSIDs did not change the results as only BSSIDs are taken into account by the WPS.

7. CONCLUSION

This paper presented an attack that could be mounted by a physical proximity-based attacker to get the identity of a target. The attack is generic in nature, i.e., all devices using vulnerable Wi-Fi-based Positioning Systems could potentially be exploited with the help of geotagged services that provide geolocation information publicly. The work demonstrated that users' privacy is at risk if proper cautions are not taken to prevent location spoofing in WPSs.

8. REFERENCES

[1] Wi-Fi tracking in retail stores. `http://www.washingtonpost.com/blogs/the-switch/wp/2013/10/19/how-stores-use-your-phones-wifi-to-track-your-shopping-habits/`, consulted on 2014.04.01.

[2] J. P. Achara, M. Cunche, V. Roca, and A. Francillon. Short Paper: WifiLeaks: Underestimated Privacy Implications of the ACCESS_WIFI_STATE Android Permission. In *7th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, Oxford, United Kingdom, July 2014.

[3] M. Cunche. I know your MAC address: Targeted tracking of individual using Wi-Fi. *Journal of Computer Virology and Hacking Techniques*, pages 1–9, 2013.

[4] M. Cunche, M. A. Kaafar, and R. Boreli. I know who you will meet this evening! linking wireless devices using Wi-Fi probe requests. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a*, pages 1–9. IEEE, 2012.

[5] Y. Gao, F. Wang, H. Luan, and T.-S. Chua. Brand data gathering from live social media streams. In *Proceedings of International Conference on Multimedia Retrieval*, ICMR '14, pages 169:169–169:176, New York, NY, USA, 2014. ACM.

[6] B. Greenstein, R. Gummadi, J. Pang, M. Y. Chen, T. Kohno, S. Seshan, and D. Wetherall. Can Ferris Bueller still have his day off? protecting privacy in the wireless era. In *Proceedings of the 11th USENIX workshop on Hot topics in operating systems*, pages 10:1–10:6, Berkeley, CA, USA, 2007.

[7] B. Henne, C. Szongott, and M. Smith. SnapMe if You Can: Privacy Threats of Other Peoples' Geo-tagged Media and What We Can Do About It. In *Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks*, WiSec '13, pages 95–106, New York, NY, USA, 2013. ACM.

[8] A. B. M. Musa and J. Eriksson. Tracking unmodified smartphones using Wi-Fi monitors. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, SenSys '12, pages 281–294, New York, NY, USA, 2012. ACM.

[9] L. Nguyen, Y. S. Kim, P. Tague, and J. Zhang. IdentityLink: User-Device Linking through Visual and RF-Signal Cues. In *ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, Sept 2014.

[10] S. Pandey and P. Agrawal. A survey on localization techniques for wireless networks. *Journal of the Chinese Institute of Engineers*, 29(7):1125–1148, 2006.

[11] D. Sanai. Detection of Promiscuous Nodes Using ARP Packets. *A white paper from http://www.securityfriday.com Accessed in Aug*, 2002.

[12] M. Srivatsa and M. Hicks. Deanonymizing mobility traces: Using a social network as a side-channel. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, Oct. 2012.

[13] N. O. Tippenhauer, K. B. Rasmussen, C. Pöpper, and S. Čapkun. Attacks on public WLAN-based positioning systems. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, pages 29–40. ACM, 2009.

[14] M. Vanhoef and F. Piessens. Advanced Wi-Fi attacks using commodity hardware. In *Proceedings of the 30th Annual Computer Security Applications Conference*, pages 256–265. ACM, 2014.

[15] Z. Yang, C. Wu, and Y. Liu. Locating in fingerprint space: Wireless indoor localization with little human intervention. In *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*, Mobicom '12, pages 269–280, New York, NY, USA, 2012. ACM.